# Shortest Connection Networks And Some Generalizations

By R. C. PRIM

(Manuscript received May 8, 1957)

*The basic problem considered is that of interconnecting a given set of terminals with a shortest possible network of direct links. Simple and practical procedures are given for solving this problem both graphically and computationally. It develops that these procedures also provide solutions for a much broader class of problems, containing other examples of practical interest.*

## I. INTRODUCTION

A problem of inherent interest in the planning of large-scale communication, distribution and transportation networks also arises in connection with the current rate structure for Bell System leased-line services. It is the following:

Basic Problem — *Given a set of (point) terminals, connect them by a network of direct terminal-to-terminal links having the smallest possible total length (sum of the link lengths).* (A *set* of terminals is "connected," of course, if and only if there is an unbroken chain of links between every two terminals in the set.) An example of such a *Shortest Connection Network* is shown in Fig. 1. The prescribed terminal set here consists of Washington and the forty-eight state capitals. The distances on the particular map used are accepted as true.

Two simple construction principles will be established below which provide simple, straight-forward and flexible procedures for solving the basic problem. Among the several alternative algorithms whose validity follows from the basic construction principles, one is particularly well adapted for automatic computation. The nature of the construction principles and of the demonstration of their validity leads quite naturally to the consideration, and solution, of a broad class of minimization problems comprising a non-trivial abstraction and generalization of the basic problem. This extended class of problems contains examples of practical

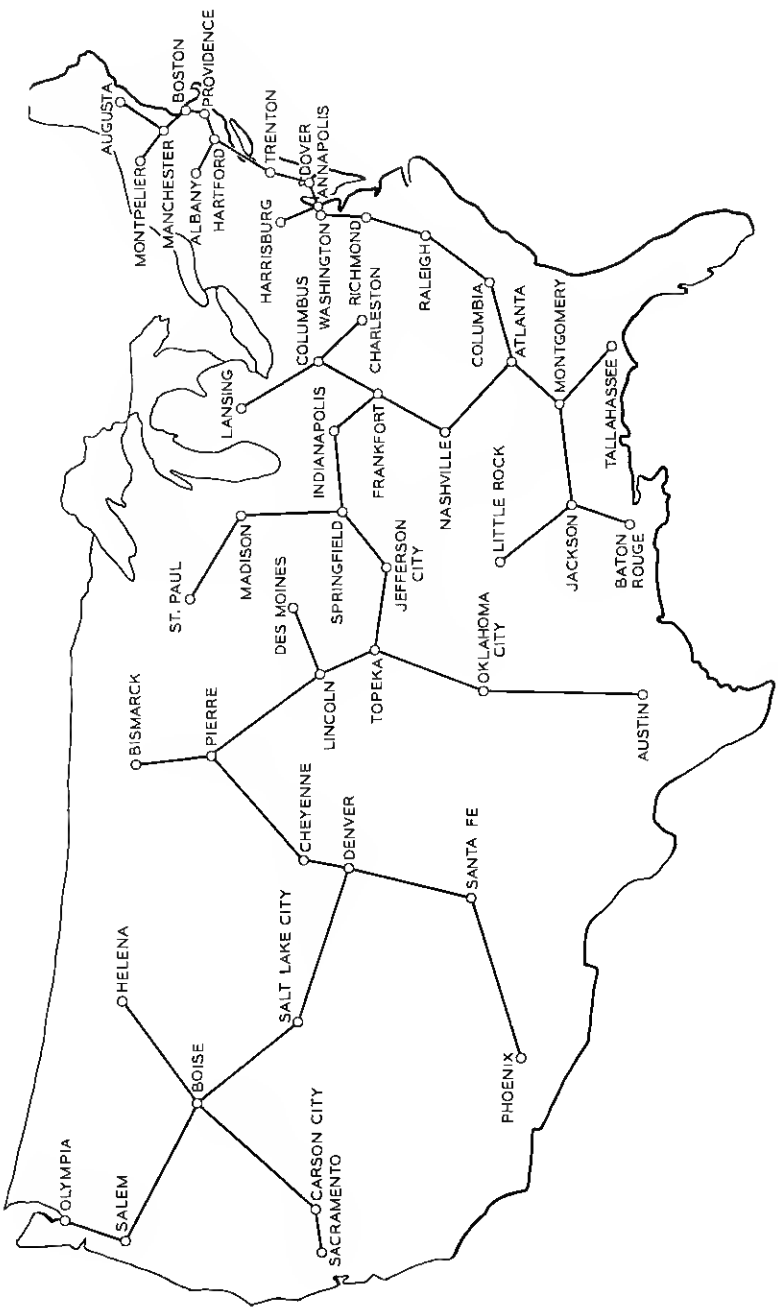


Fig. 1 — Example of a shortest connection network.

interest in quite different contexts from those in which the basic problem had its genesis.

## II. CONSTRUCTION PRINCIPLES FOR SHORTEST CONNECTION NETWORKS

In order to state the rules for solution of the basic problem concisely, it is necessary to introduce a few, almost self-explanatory, terms. An *isolated terminal* is a terminal to which, at a given stage of the construction, no connections have yet been made. (In Fig. 2, terminals 2, 4, and 9 are the only isolated ones.) A *fragment* is a terminal subset connected by direct links, between members of the subset. (In Fig. 2, 8-3, 1-6-7-5, 5-6-7, and 1-6 are some of the fragments; 2-4, 4-8-3, 1-5-7, and 1-7 are

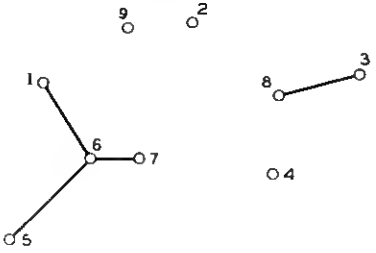


Fig. 2 — Partial connection network.

not fragments.) The *distance of a terminal from a fragment* of which it is not an element is the minimum of its distances from the individual terminals comprising the fragment. An *isolated fragment* is a fragment to which, at a given stage of the construction, no external connections have been made. (In Fig. 2, 8-3 and 1-6-7-5 are the only isolated fragments.) A *nearest neighbor of a terminal* is a terminal whose distance from the specified terminal is at least as small as that of any other. A *nearest neighbor of a fragment*, analogously, is a terminal whose distance from the specified fragment is at least as small as that of any other.

The two fundamental construction principles (P1 and P2) for shortest connection networks can now be stated as follows:

Principle 1 — *Any isolated terminal can be connected to a nearest neighbor.*

Principle 2 — *Any isolated fragment can be connected to a nearest neighbor by a shortest available link.*

For example, the next steps in the incomplete construction of Fig. 2 could be any one of the following:

(1) add link 9-2 (P1 applied to Term. 9)

(2) add link 2-9 (P1 applied to Term. 2)
(3) add link 4-8 (P1 applied to Term. 4)
(4) add link 8-4 (P2 applied to frag. 3-8)
(5) add link 1-9 (P2 applied to frag. 1-6-7-5).

One possible sequence for completing this construction is: 4-8 (P1), 8-2 (P2), 9-2 (P1), and 1-9 (P2). Another is: 1-9 (P2), 9-2 (P2), 2-8 (P2), and 8-4 (P2).

As a second example, the construction of the network of Fig. 1 could have proceeded as follows: Olympia-Salem (P1), Salem-Boise (P2), Boise-Salt Lake City (P2), Helena-Boise (P1), Sacramento-Carson City (P1), Carson City-Boise (P2), Salt Lake City-Denver (P2), Phoenix-Santa Fe (P1), Santa Fe-Denver (P2), and so on.

The kind of intermixture of applications of P1 and P2 demonstrated here is very efficient when the shortest connection network is actually being laid out on a map on which the given terminal set is plotted to scale. With only a few minutes of practice, an example as complex as that of Fig. 1 can be solved in less than 10 minutes. Another mode of procedure, making less use of the flexibility permitted by the construction principles, involves using P1 only once to produce a single fragment, which is then extended by successive applications of P2 until the network is completed. This highly systematic variant, as will emerge later, has advantages for computer mechanization of the solution process. As applied to the example of Fig. 1, this algorithm would proceed as follows if Sacramento were the indicated initial terminal: Sacramento-Carson City, Carson City-Boise, Boise-Salt Lake City, Boise-Helena, Boise-Salem, Salem-Olympia, Salt Lake City-Denver, Denver-Cheyenne, Denver-Santa Fe, and so on.

Since each application of either P1 or P2 reduces the total number of isolated terminals and fragments by one, it is evident that an $N$-terminal network is connected by $N$-1 applications.

## III. VALIDATION OF CONSTRUCTION PRINCIPLES

The validity of P1 and P2 depends essentially on the establishment of two *necessary* conditions (NC1 and NC2) for a *shortest* connection network (SCN):

Necessary Condition 1 — *Every terminal in a SCN is directly connected to at least one nearest neighbor.*

Necessary Condition 2 — *Every fragment in a SCN is connected to at least one nearest neighbor by a shortest available path.*

To simplify the argument, it will at first be assumed that all distances

between terminals are different, so that each terminal or fragment has a single, uniquely defined, nearest neighbor. This restriction will be removed later.

Consider first NC1. Suppose there is a SCN for which it is untrue. Then [Fig. 3(a)] some terminal, $t$, in this network is not directly joined to its nearest neighbor, $n$. Since the network is connected, $t$ is necessarily joined directly to some one or more terminals other than $n$ — say $f_1, \cdots, f_r$. For the same reason, $n$ is necessarily joined through some chain, $C$, of one or more links to one of $f_1, \cdots, f_r$ — say to $f_k$. Now if the link $t - f_k$ is removed from the network and the link $t - n$ is added [Fig. 3(b)], the connectedness of the network is clearly not destroyed — $f_k$ being joined to $t$ through $n$ and $C$, rather than directly. However, the total length of the network has now been *decreased*, because, by hypothesis, $t - n$ is shorter than $t - f_k$. Hence, contrary to the initial supposition, the network contradicting NC1 could not have been the *shortest*, and the truth of NC1 follows. From NC1 follows P1, which merely *permits* the addition of links which NC1 shows *have* to appear in the final SCN.

Turning now to NC2, the above argument carries over directly if $t$ is thought of as a fragment of the supposed contradictory SCN, rather than as an individual terminal — provided, of course that the $t - n$ link substituted for $t - f_k$ is the shortest link from $n$ to any of the terminals belonging to $t$. From the validity of NC2 follows P2 — again the links whose addition is permitted by P2 are all necessary, by NC2, in the final SCN.

The temporary restrictive assumption that no two inter-terminal distances are identical must now be removed. A reappraisal of the proofs of NC1 and NC2 shows that they are still valid if $n$ is not the only terminal at distance $t - n$ from $t$, for in the supposedly contradictory network the distance $t - f_k$ must be *greater* than $t - n$. What remains to be established is that the length of the final connection network resulting

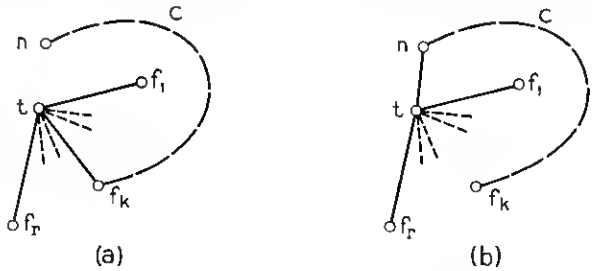


Fig. 3 — Schematic demonstration of NC1.

from successive applications ($N-1$ for $N$ terminals) of P1 and P2 is independent of *which* nearest neighbor is chosen for connection at a stage when more than one nearest neighbor to an isolated terminal or

t is available. This is a consequence of the following considerations: For a prescribed terminal set there are only a *finite* number of connection networks (certainly fewer than $C_{N-1}^{N(N-1)/2}$ — the number of distinct ways of choosing $N-1$ links from the total of $N(N-1)/2$ possible links). The length of each one of this finite set of connection networks is a *continuous* function of the individual interterminal distances, $d_{ij}$ (as a matter of f ct, it is a linear function with coefficients 0 and 1). With the $d_{ij}$ specified, the length, $L$, of a shortest connection network is simply the smallest length in this finite set of connection network lengths. Therefore $L$ is uniquely determined. (It may, of course, be associated with more than one of the connection networks.) Now, if at each stage of construction employing P1 and P2 at which a choice is to be made among two or more nearest neighbors $n_1, \cdots, n_r$ of an isolated terminal (or fragment) $t$, a small positive quantity, $\epsilon$, is subtracted from any specific one of the distances $d_{tn_1}, \cdots, d_{tn_r}$ — say from $d_{tn_k}$ — the construction will be uniquely determined. The total length, $L'$, of the resulting SCN for the modified problem will now depend on $\epsilon$, as well as on the $d_{ij}$ of the original terminal set. The dependence on $\epsilon$ will be *continuous*, however, because the minimum of a finite set of continuous functions of $\epsilon$ (the set of lengths of all connection networks of the modified problem) is itself a continuous function of $\epsilon$. Hence, as $\epsilon$ is made vanishingly small, $L'$ approaches $L$, regardless of which "nearest neighbor" links were chosen for shortening to decide the construction.

## IV. ABSTRACTION AND GENERALIZATION

In the examples of Figs. 1 and 2, the terminal set to be connected was represented by points on a distance-true map. The decisions involved in applying P1 and P2 could then be based on visual judgements of relative distances, perhaps augmented by application of a pair of dividers in a few close instances. These direct geometric comparisons can of course, be replaced by numerical ones if the inter-terminal distances are entered on the terminal plot, as in Fig. 4(a). The application of P1 and P2 goes through as before, with the relevant "nearest neighbors" determined by a comparison of numerical labels, rather than by a geometric scanning process. For example, P1 applied to Terminal 5 of Fig. 4(a) yields the Link 5-6 of the SCN of Fig. 4(b), because 4.6 is less than 5.6, 8.0, 8.5, and 5.1, and so on.

When the construction of shortest connection networks is thus reduced to processes involving only the numerical distance labels on the various possible links, the *actual location* of the points representing the various terminals in a graphical representation of the problem is, of course, inconsequential. The problem of Fig. 4(a) can just as well be represented by Fig. 5(a), for example, and P1 and P2 applied to obtain the SCN of Fig. 5(b). The original metric problem concerning a set of points in the plane has now been abstracted into a problem concerning *labelled graphs*. The correspondence between the terminology employed thus far and more conventional language of Graph Theory is as follows:

terminal ↔ vertex
possible link ↔ edge
length of link ↔ "length" (or "weight") of edge
connection network ↔ spanning subgraph
(without closed loops) ↔ (spanning subtree)

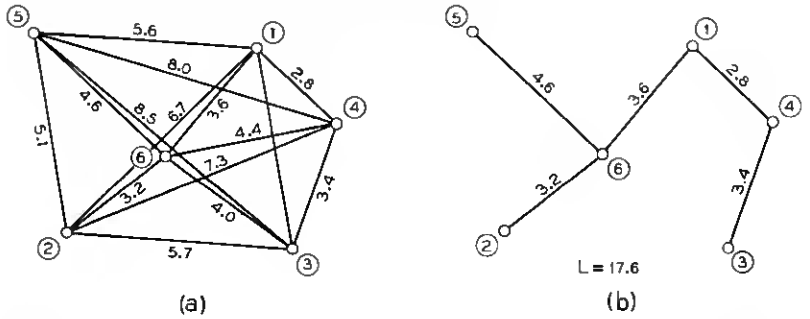


Fig. 4 — Example of a shortest spanning subtree of a complete labelled graph.

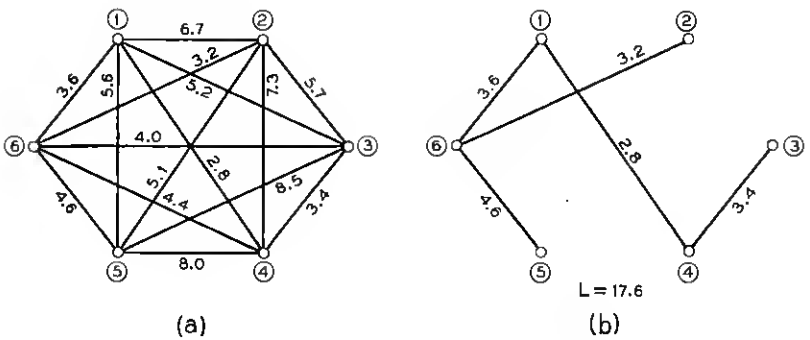


Fig. 5 — Example of a shortest spanning subtree of a complete labelled graph.

shortest connection network ↔ shortest spanning subtree

SCN ↔ SSS

It will be useful and worthwhile to carry over the concepts of "fragment" and "nearest neighbor" into the graph theoretic framework. P1 and P2 can now be regarded as construction principles for finding a shortest spanning subtree of a labelled graph.

In the originating context of connection networks, the graphs from which a shortest spanning subtree is to be extracted are *complete graphs*; that is, graphs having an edge between every pair of vertices. It is natural, now, to generalize the original problem by seeking shortest spanning subtrees for arbitrary connected labelled graphs. Consider, for example, the labelled graph of Fig. 6(a) which is derived from that of Fig. 5(a) by deleting some of the edges. (In the connection network context, this is equivalent to barring direct connections between certain terminal pairs.) It is easily verified that NC1 and NC2, and hence P1 and P2, are valid also in these more general cases. For the example of Fig. 6(a), they yield readily the SSS of Fig. 6(b).

As a further generalization, it is not at all necessary for the validity of P1 aud P2 that the edge "lengths" in the given labelled graph be derived, as were those of Figs. 4–6, from the inter-point distances of some point set in the plane. *P1 and P2 will provide a SSS for any connected labelled graph with any set of real edge "lengths."* The "lengths" need not even be positive, or of the same sign. See, for example, the labelled graph of Fig. 7(a) and its SSS, Fig. 7(b). It might be noted in passing that this degree of generality is sufficient to include, among other things, shortest connection networks in an arbitrary number of dimensions.

A further extension of the range of problems solved by P1 and P2 follows trivially from the observation that the *maximum* of a set of

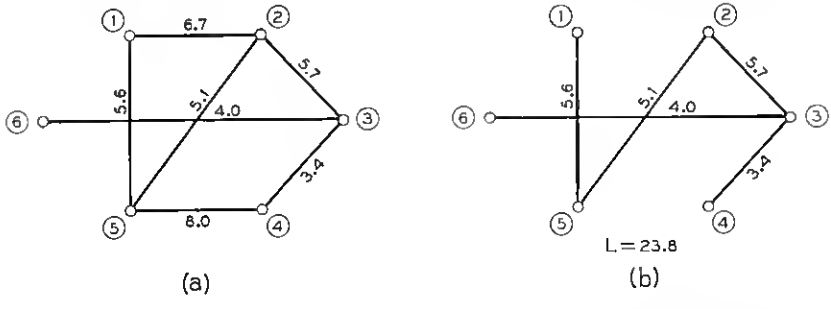


Fig. 6 — Example of a shortest spanning subtree of an incomplete labelled graph.

real numbers is the same as the negative of the *minimum* of the negatives of the set. Therefore, P1 and P2 can be used to construct a *longest spanning subtree* by changing the signs of the "lengths" on the given labelled graph. Fig. 8 gives, as an example, the longest spanning subtree for the labelled graph of Figs. 4(a) and 5(a).

It is easy to extend the arguments in support of NC1 and NC2 from the simple case of minimizing the sum to the more general problems of minimizing an arbitrary increasing symmetric function, or of maximizing an arbitrary decreasing symmetric function, of the edge "lengths" of a spanning subtree. (A symmetric function of $n$ variables is one whose value is unchanged by any interchanges of the variable values; e.g., $x_1 + x_2 + \cdots + x_n$, $x_1 x_2 \cdots x_n$, $\sin 2x_1 + \sin 2x_2 + \cdots + \sin 2x_n$, $(x_1^3 + x_2^3 + \cdots + x_n^3)^{1/2}$, etc.) From this follow the non-trivial generalizations.

> The *shortest spanning subtree* of a connected labelled graph also minimizes all increasing symmetric functions, and maximizes all decreasing symmetric functions, of the edge "lengths."

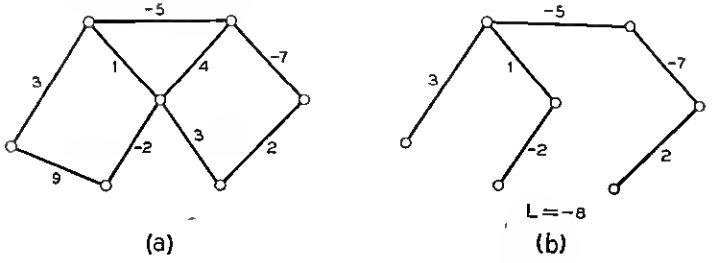


Fig. 7 — Example of a shortest spanning subtree of a labelled graph with some "lengths" negative.

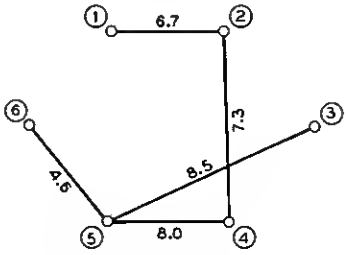


Fig. 8 — The longest spanning subtree of the labeled graph of Figs. 4(a) and 5(a).

The *longest spanning subtree* of a connected labelled graph also maximizes all increasing symmetric functions, and minimizes all decreasing symmetric functions, of the edge "lengths."

For example, with positive "lengths" the same spanning subtree that minimizes the *sum* of the edge "lengths" also minimizes the *product* and the *square root of the sum of the squares*. At the same time, it maximizes the sum of the reciprocals and the product of the arc cotangents.

It seems likely that these extensions of the original class of problems soluble by P1 and P2 contain many examples of practical interest in quite different contexts from the original connection networks. A not entirely facetious example is the following: A message is to be passed to all members of a certain underground organization. Each member knows some of the other members and has procedures for arranging a rendezvous with anyone he knows. Associated with each such possible rendezvous — say between member "$i$" and member "$j$" — is a certain probability, $p_{ij}$, that the message will fall into hostile hands. How is the message to be distributed so as to minimize the over-all chances of its being compromised? If members are represented as vertices, possible rendezvous as edges, and compromise probabilities as "length" labels in a labelled graph, the problem is to find a spanning subtree for which $1 - \Pi(1 - p_{ij})$ is minimized. Since this is an increasing symmetric function of the $p_{ij}$'s, this is the same as the spanning subtree minimizing $\Sigma\, p_{ij}$, and this is easily found by P1 and P2.

Another application, closer to the original one, is that of minimizing the lengths of wire used in cabling panels of electrical equipment. Restrictions on the permitted wiring patterns lead to shortest connection network problems in which the effective distances between terminals are not the direct terminal-to-terminal distances. Thus the more general viewpoint of the present section is applicable.

## V. COMPUTATIONAL TECHNIQUE

Return now to the exemplary shortest connection network problem of Figs. 4(a) and 5(a) which served as the center for discussion of the arithmetizing of the metric factors in the Basic Problem. It is evident that the actual drawing and labelling of all the edges of a complete graph will get very cumbersome as the number of vertices increases — an $N$-vertex graph has $(1/2)(N^2 - N)$ edges. For large $N$, it is convenient to organize the numerical metric information in the form of a *distance table*, such as Fig. 9, which is equivalent in content to Fig. 4(a) or Fig.

5(a). (A distance table can also be prepared to represent an incomplete labelled graph by entering the length of non-existent edges as $\infty$.)

When it is desired to determine a shortest connection network directly from the distance table representation — either manually, or by machine computation — one of the numerous particular algorithms obtainable

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | — | 6.7 | 5.2 | 2.8 | 5.6 | 3.6 |
| 2 | 6.7 | — | 5.7 | 7.3 | 5.1 | 3.2 |
| 3 | 5.2 | 5.7 | — | 3.4 | 8.5 | 4.0 |
| 4 | 2.8 | 7.3 | 3.4 | — | 8.0 | 4.4 |
| 5 | 5.6 | 5.1 | 8.5 | 8.0 | — | 4.6 |
| 6 | 3.6 | 3.2 | 4.0 | 4.4 | 4.6 | — |

Fig. 9 — Distance table equivalent of Figs. 4(a) and 5(a).

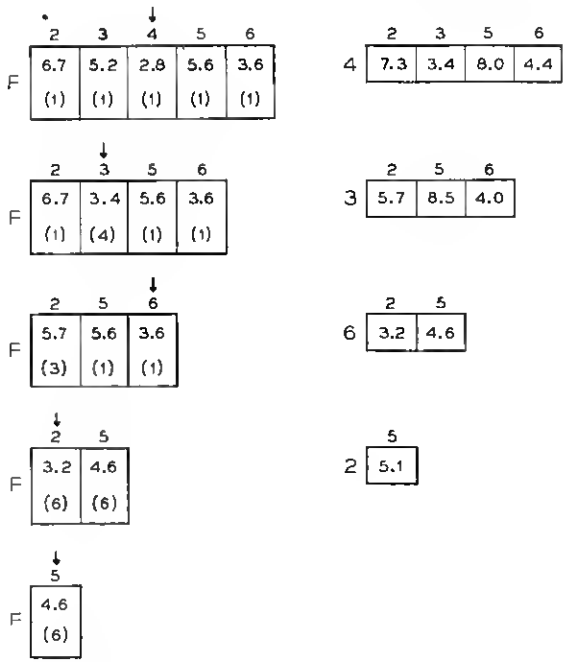


Fig. 10 — Illustrative computational determination of a shortest connection network.

by restricting the freedom of choice allowed by P1 and P2 is distinctly superior to other alternatives. This variant is the one in which P1 is used but once to produce a single isolated fragment, which is then extended by repeated applications of P2.

The successive steps of an efficient computational procedure, as applied to the example of Fig. 9, are shown in Fig. 10. The entries in the top rows of the successive $F$ tables are the distances from *the* connected fragment to the unconnected terminals at each stage of fragment growth. The entries in parentheses in the second rows of these tables indicate the nearest neighbor *in the fragment* of the external terminal in question. The computation is started by entering the first row of the distance table into the $F$ table (to start the growing fragment from Terminal 1). A smallest entry in the $F$ table is then selected (in this case, 2.8, associated with External Terminal 4 and Internal Terminal 1). The link 1-4 is deleted from the $F$ table and entered in the Solution Summary (Fig. 11). The remaining entries in the first stage $F$ table are then compared with the corresponding entries in the "4" row of the distance table (reproduced beside the first $F$ table). If any entry of this "added terminal" distance table is smaller than the corresponding $F$ table entry, it is substituted for it, with a corresponding change in the parenthesized index. (Since 3.4 is less than 5.2, the 3 column of the $F$ table becomes 3.4/(4).) This process is repeated to yield the list of successive nearest neighbors to the growing fragment, as entered in Fig. 11. The $F$ and "added terminal" distance tables grow shorter as the number of unconnected terminals is decreased.

This computational procedure is easily programmed for an automatic computer so as to handle quite large-scale problems. One of its advantages is its avoidance of checks for closed cycles and connectedness. Another is that it never requires access to more than two rows of distance data at a time — no matter how large the problem.

SOLUTION SUMMARY

| LINK | LENGTH |
|---|---|
| 1 – 4 | 2.8 |
| 4 – 3 | 3.4 |
| 1 – 6 | 3.6 |
| 6 – 2 | 3.2 |
| 6 – 5 | 4.6 |

Fig. 11 — Solution summary for computation of Fig. 10.

## VI. RELATED LITERATURE AND PROBLEMS

J. B. Kruskal, Jr.[1] discusses the problem of constructing shortest spanning subtrees for labelled graphs. He considers only distinct and positive sets of edge lengths, and is primarily interested in establishing uniqueness under these conditions. (This follows immediately from NC1 and NC2.) He also, however, gives three different constructions, or algorithms, for finding SSS's. Two of these are contained as special cases in P1 – P2. The third is — "Perform the following step as many times as possible: Among the edges not yet chosen, choose the longest edge whose removal will not disconnect them" While this is not directly a special case of P1 – P2, it is an obvious corollary of the special case in which the *shortest* of the edges permitted by P1 – P2 is selected at each stage. Kruskal refers to an obscure Czech paper[2] as giving a construction and uniqueness proof inferior to his.

The simplicity and power of the solution afforded by P1 and P2 for the Basic Problem of the present paper comes as something of a surprise, because there are well-known problems which *seem* quite similar in nature for which no efficient solution procedure is known.

One of these is *Steiner's Problem*: Find a shortest connection network for a given terminal set, with freedom to add additional terminals wherever desired. A number of necessary properties of these networks are known[3] but do not lead to an effective solution procedure.

Another is the *Traveling Salesman Problem*: Find a closed path of minimum length connecting a prescribed terminal set. Nothing even approaching an effective solution procedure for this problem is now known (early 1957).

## REFERENCES

1. J. B. Kruskal, Jr., On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem, Proc. Amer. Math. Soc. **7**, pp. 48–50, 1956.
2. Otakar Borůvka, On a Minimal Problem, Práce Moravské Pridovedecké Spdecnosti, **3**, 1926.
3. R. Courant and H. Robbins, *What is Mathematics*, 4th edition, Oxford Univ. Press, N. Y., 1941, pp. 374 *et* seq.